

AVS: Visualisation Software for Multiple Platforms

Ian J. Curington
Senior Systems Engineer
Stardent Computer Ltd.
UK

AVS, a flexible interactive development environment for Scientific Visualisation, has recently been extended to control data flow and remote execution on networked computers. Examples of Supercomputers and Visualisation machines are described.

Ian Curington is a member of the technical staff of Stardent Computer Ltd, where he specialises in interactive visualization systems and software, and visualization consultancy. Before joining Stardent, he was Operations Director at Amazing Array productions, where he developed radiosity rendering and advanced animation software tools, and was involved in several award winning animations. From 1979 to 1987 he was signal processing, image processing specialist with Floating Point Systems, where he developed raytracing and graphics algorithms for array processors. His research interests include visualisation of fluid flows, radiosity, and vector and parallel computing.

COMPUTER GRAPHICS

COMPUTER GRAPHICS

INTERNATIONAL EXHIBITION AND CONFERENCE, 6-8 NOVEMBER 1990, ALEXANDRA PALACE, LONDON

What is AVS?

AVS (Application Visualization System) is a software system for developing interactive scientific visualisation applications quickly, with a minimum of programming effort. AVS is an application framework designed for scientists and engineers. The system is used to make applications that combine interactive graphics and high computational requirements easier to develop for both programmers and non-programmers.

AVS allows flow networks of small modules to be linked using a visual programming method [2], called the "network editor". The creation of networks of modules also provides for an automatic interactive user interface system. Modules are connected such that scientific data flows through them at run time, as specified by the flow network. Modules may be resident on any computer system on a network where AVS is installed.

Background

The massive quantities of data generated by today's supercomputers present a challenge to researchers in the assimilation and understanding of results. While graphics and computers have made rapid performance improvements in recent years, the software available to researchers and engineers has not kept pace.

Today, visualisation software has fallen into two categories: graphics subroutine libraries and animation packages. Libraries represent the traditional, structured language approach, and assume the user is a proficient programmer. While providing powerful functions, they fail to hide the basic complexity of the visualisation problem, since the programmer must understand the graphics primitives, data structures, and many other system dependent calls, to build an application. With recent exposure to easy to use PC packages, many engineers and scientists have little interest in developing new visualisation tools themselves, which may involve a large investment of time in learning a new system peripheral to the primary area of investigation.

Animation systems, on the other hand, are used as a post-process after a numerical simulation is run on a supercomputer or mini-supercomputer, typically in batch mode, and after the data has been transferred into a form acceptable by the animation system. For many packages, the capabilities are fixed, and may not match a particular discipline. They may also be restricted to graphical data structures and file formats, rather than scientific ones. Most of the high quality animation systems are monolithic, and are not based in layered industry standards such as PHIGS-Plus, X-windows, or UNIX XDR distributed network standards. In addition, the origins of animation packages in the advertising and entertainment industries orient them toward very high quality image rendering, often at the expense of interactive performance.

The AVS Modular Framework

AVS is a framework that can be used to develop scientific visualisation applications based on the concept of combining interactive visualisation techniques with research and engineering computing [3,4]. It is designed to be used by scientists and engineers, without assistance, rather than software developers. The underlying flexibility, however, means that AVS can be used as an embedded tool kit by package developers. The general characteristics of the AVS software system can be summarized as:

- * **Ease of Use.** By employing interactive user interfaces and simplifying the programmer's task, the system is more accessible to scientists and engineers. By exploiting data structure commonality, application development that takes weeks of programming by library systems is often reduced to only a few hours of direct prototyping with AVS.
- * **Low Cost.** In contrast to large fixed-function packages, AVS is a framework that integrates small scale software components (modules), which are much less costly to develop and maintain.
- * **Completeness.** Unlike other visualisation software which focuses on graphics rendering and viewing, AVS spans the whole process from extracting data from a simulation or experiment, filtering, enhancement, defining representation, as well as rendering and display.
- * **Extensibility.** AVS assumes that scientists and engineers will need to extend and customise the visualisation process for individual needs. All flow networks may be edited graphically by end users, and full source code is available for most modules. A module developers guide is available, and courses are held on a regular basis.
- * **Portability.** To be truly useful, software should easily move to new systems, particularly as hardware evolves in performance and capability. AVS is built on standard layers such as PHIGS+, Dore, X11 Window System, UNIX, and network protocols for remote module control, to enable it to have a long life. As this paper is being written AVS has already been licensed by DIGITAL, Evans and Sutherland, Tektronix, CONVEX, and FPS Computing, and demonstrated on their respective systems, and of course AVS is available on all Stardent systems.

AVS Now On CRAY Research Supercomputers

AVS3 is the next release of Stardent's Application Visualization System. Concurrent with the announcement of AVS3 in Europe, Stardent is announcing the porting of AVS3 to the CRAY RESEARCH family of Supercomputers.

The use of AVS on a distributed network of Stardent and Cray Research machines enables both graphical and compute intensive processes to be distributed and controlled in one interactive window based environment. In previous distributed data visualization systems a rigid partitioning of the scientific application and the graphics visualization process was required. However, AVS allows users to customise at run time the number and character of processes that run across the combined network of Stardent and Cray Research Computers. A typical data flow diagram is shown at the end of the paper.

The network editor provides a very easy way of making new flow networks, with no need to refer to a manual. Modules are "dragged" from a module list over to a worksheet, and connected by a simple on-screen point and click method. Modules can be connected to each other in any order, with automatic data type checking performed as connections are made. The data connections are shown immediately as they are made, and can be used immediately, with no other operations required. Running networks can be edited to dynamically add, connect, or delete modules. Modules are only re-executed when new data is present, or a user parameter has changed.

The editor includes menu controls to specify which computational modules are running on the CRAY and which run on the Stardent. In this way, multiple machine execution is a natural extension of the AVS network philosophy, and uses the network editor in exactly the same way.

The CRAY resident AVS modules have direct access to the interactive X-Window based user interface services on the Stardent systems. In this way, local and remote modules behave in exactly the same way, and have identical user interfaces. The user is completely free to choose where a module executes, given that the same module is on each system.

Data transfer between the Stardent and CRAY machines is handled through an External Data Representation (XDR) mechanism which provides total transparency during the data flow and automatically compensates for the machine specific data formats. The conversion mechanism is automatically invoked by AVS, when the flow network crosses machine boundaries.

The recent announcement of AVS support for 8 bit colour X-terminals and workstations, running in conjunction with Stardent systems, enables the user to choose between the high performance State Of The Art Stardent system Visualization graphics and lower cost devices, using the Stardent system as the 'AVS Server", and gateway to the Cray via AVS. This development permits more low cost seats access to the Stardent/Cray AVS environment.

The first delivery configurations will require a Stardent system running in conjunction with a CRAY Research YMP (UNICOS) with other CRAY models such XMP and CRAY-2 to follow.

The multiplicity of benefits from this unique new capability provides the user with visualization access to data resident on high speed CRAY file systems and data generated dynamically by CRAY computations. The portions of AVS resident on the CRAY allow for data extraction, enhancement, and mapping operations (using a module library), with the Stardent machine providing filtering, mapping, graphical rendering, and display functions. It is expected that the distributed version of AVS running on the CRAY and Stardent systems will be a significant factor in supercomputing strategy for many large CRAY sites [1].

Summary

The modular and extensible software design of AVS has enabled it to be extended to include multiple-machine data flow networks, linking CRAY Supercomputers and Stardent systems together into one interactive heterogeneous visualisation environment. Visual system configuration and data flow specification further enhances the accessibility of AVS to non-programmer scientists and engineers.

References:

- [1] Cabral and Hunter, "Visualization Tools at Lawrence Livermore National Laboratory", Computer (IEEE) August 1989.
- [2] Myers, "Taxonomies of Visual Programming and Program Visualization", Journal of Visual Languages and Computing, Vol 1, No. 1, Academic Press Ltd., 1990.
- [3] Upson, Craig, "The Visualisation of Volumetric Data", Computer Graphics '89, Blenheim Online Publications, London, November 1989.
- [4] Upson, etal., "The Application Visualization System: A Computational Environment for Scientific Visualization", IEEE Computer Graphics and Applications, July 1989.

Example Network

